Programmation orientée objet (C++) – ING2-GSI

TP5 – 1 : Relation entre classes

La classe Segment

Objectifs

- Comprendre les différentes relations entre classes
- Comprendre comment utiliser un objet composant
- Faire la différence entre les relations d'agrégation et de composition

Ce TP propose de développer une classe Segment (et une classe Cercle pour les plus rapides ou motivés).

Exercice 5.1 : Définition d'une classe Segment

L'objectif de cet exercice est de comprendre, au travers de la définition d'un segment, les relations entre classes, en particulier la relation d'agrégation et la relation de composition.

Un segment est défini par la donnée de ses deux points extrémités. Il a une longueur et peut être affiché et translaté en précisant un déplacement suivant l'axe des X (abscisses) et un déplacement suivant l'axe des Y (ordonnées).

Remarque : On considère que nous disposons de la classe Point (fichiers fournis).

5.1.1. *Modélisation UML*. Proposer un diagramme de classes UML qui décrit le segment.

5.1.2. *Type de relation entre Segment et Point*. Lorsqu'une requête d'une classe a pour type une autre classe, on préfère faire apparaître explicitement une relation entre ces deux classes. Le couplage entre les deux classes est précisé grâce au type de relation : association, agrégation ou composition.

L'association est la relation la plus générale. Les relations d'agrégation et de composition en sont des spécialisations. Elles ont une sémantique plus précise. Elles correspondent à la relation tout/partie.

Ici, la relation entre segment et point est une relation tout ou partie, le segment est le tout et le point est une partie du segment. Il s'agit donc soit d'agrégation, soit de composition.

Dans le cas de l'agrégation, les instances de Point et de Segment ont des durées de vie différentes. Un même point peut donc intervenir dans la constitution de plusieurs segments et, bien sûr, sa durée de vie sera supérieure à celle des deux segments.

Dans le cas de la composition, les extrémités sont propres à chaque segment et ne peuvent donc pas être partagées. En particulier, la durée de vie des extrémités du segment est la même que celle du segment lui-même : elles sont créées avec le segment et détruites avec lui.

Pour bien comprendre la différence entre agrégation et composition, prenons l'exemple d'un avion, d'une tour de contrôle et des ailes de l'avion. Il y a relation de composition entre l'avion et ses ailes (si l'avion s'écrase, ses ailes sont détruites). En revanche, il y a relation d'agrégation – en fait association – entre l'avion et la tour de contrôle qui est utilisée par plusieurs avions. Remarquons que si l'avion s'écrase, la tour de contrôle ne sera pas détruite (sauf s'il s'écrase dessus !).

Quelle est la relation qui lie un segment à ses extrémités ?

Indication : Considérons un exemple avec deux segments s1 et s2 qui ont une extrémité initialisée à partir d'un même point p2.

```
Point p1(1,1), p2(2,2), p3(3,1);
Segment s1(p1, p2), s2(p2, p3);
s1.translater(1, 0);
```

Quel est le résultat de la translation du segment s1 (figure 1) ?

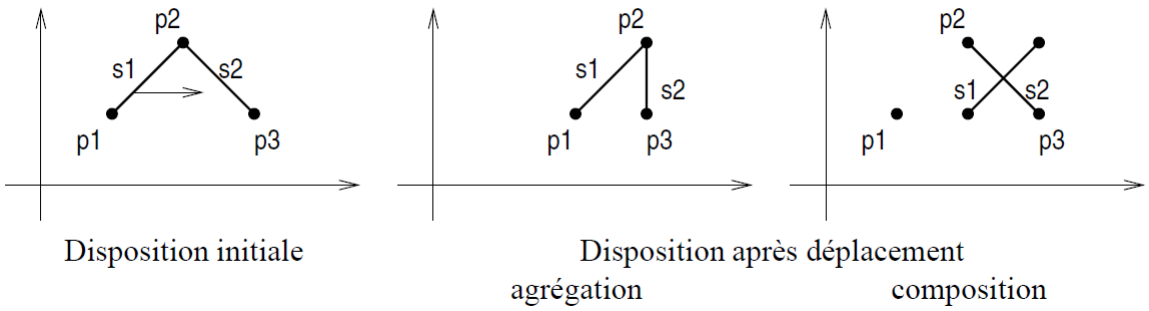

Figure 1. Dispositions possibles des deux segments après s1.translater(1,0)

5.1.3. *SegmentComposant :* les points comme composants du segment. Par analogie avec la classe Point, on peut définir les extrémités par les deux attributs suivants.

```
Point _e1, _e2; // Les deux extrémités du segment
```

Les points extrémités peuvent être considérés comme embarqués dans le segment.

5.1.3.1. Écrire la classe SegmentComposant.

5.1.3.2. Écrire un programme "*testSegmentComposant.cpp*" qui permet de vérifier la classe *SegmentComposant* ainsi écrite.

5.1.4. *SegmentPointeur :* les points définis comme pointeurs. Au lieu de définir, les extrémités comme des composants, on peut les définir par des pointeurs.

```
Point *_e1, *_e2; // Les deux extrémités du segment
```

5.1.4.1. Écrire la classe SegmentPointeur.

5.1.4.2. Écrire un programme "*testSegmentPointeur.cpp*" qui permet de vérifier la classe *SegmentPointeur* ainsi écrite.

3.1.5. Conclusions. Que peut-on conclure sur les deux solutions proposées pour définir la classe Segment ?

Exercice 5.2 : Définition d'une classe Cercle

Définir une classe Cercle. Un cercle est caractérisé par son centre et son rayon. Un cercle peut être initialisé soit à partir de son centre et de son rayon, soit à partir de deux points diamétralement opposés. Un cercle possède un périmètre et une surface. Il peut être affiché, translaté, son diamètre modifié ainsi que sa couleur.

5.2.1. Dessiner le diagramme de classes incluant la classe Cercle.

5.2.2. Écrire et tester la classe Cercle.

5.2.3. Peut-on définir un cercle à partir de son centre et d'un point de sa circonférence ?